Daniil Saltykov
09.06.2024

Course Project Report

Introduction

Our task was to develop a project that consists of two major programs. Part A was aimed to enhance our programming skills and demonstrate Python skills, importing data from a text file, processing it, calculating some relevant statistics, exporting the data to another text data file and visualizing results through graphs. Part B was aiming to get to know an artificial neural network, learn how to train it, test it, and perform some statistics based on the results and visualize results using graphs. Both parts were implemented using multiple python libraries such as MatplotLib, Pandas, Numpy, Scikit-learn and etc. Both parts were using the menu style to helpfully navigate the user throughout the capabilities of the programs.

Description

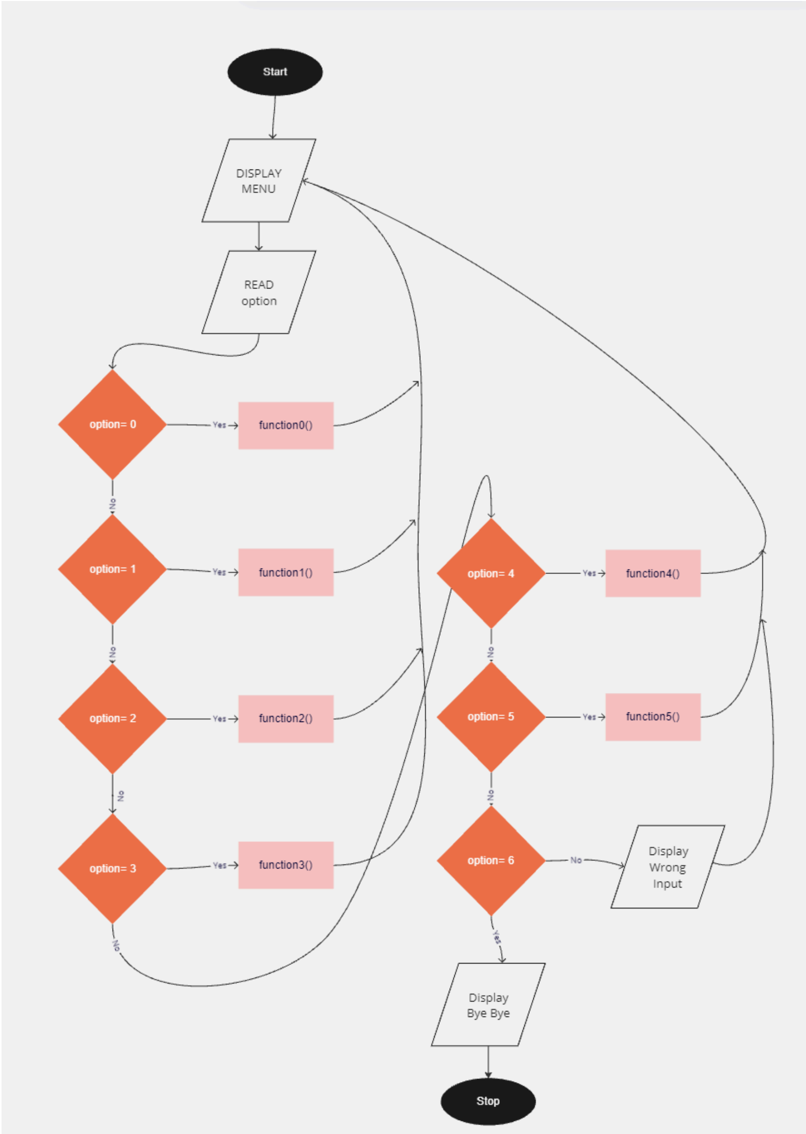
As was mentioned before, the whole program for both cases is a while loop that keeps the menu working.
Inside this loop is the match case method, which runs different parts of the code for different utilities, depending on the menu item selected by the user. At the start of a program, the user can see the greeting and the menu with the valuable options.

```
"C:\Users\Asus\Desktop\AI shit py\AI_project\.venv\Scripts\python.exe" "C:\Users\Asus\Desktop\AI shit py\AI_pro
=================================================================
F1 GRAND PRIX RACING DATA & STATISTICS FOR THE 2023 RACING SEASON
=================================================================
-------------------------------------------------------------------------------------------------


1. Read and display the F1 Grand Prix data for the 2023 racing season
2. Filter and sort race data based on a minimum threshold of laps
3. Calculate average lap time per race, save, retrieve, display
4. Sort and display the data based on user parameters
5. Calculate and graph total lap time per driver
6. Exit the program
Hint: Enter "0" for more detailed description of each menu option.

Enter your choice: |
```

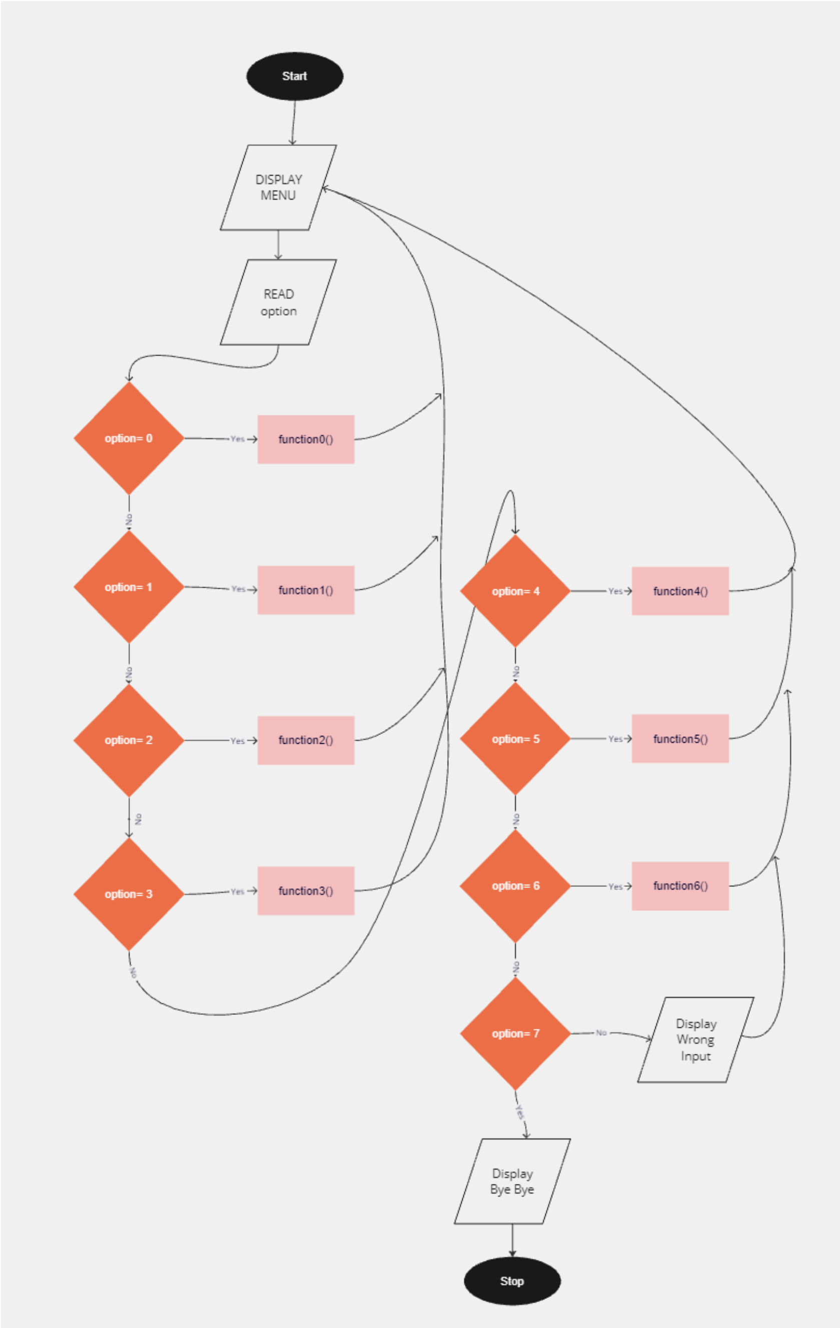
Picture 1 - Start of the program and main menu

After the input of the menu item, the program calls the corresponding code snippet that releases the selected utility. After all of the work is done, the program waits for the user to press any key and then returns to the main menu and asks the user for another selection.

The top-level flowchart for the Part A and Part B are presented below:



Picture 2 - Flowchart Part A



Picture 3 - Flowchart Part B

If the user enters “0”, the whole description of each utility will appear.

```
Enter your choice: 0
--------------------------------------------------------------------------------------------------


1. Reads the 6 columns of data from file partA_input_data.txt and neatly displays it on screen.
2. Asking user for a limit of laps to search by, then displays only the race results
which involve that number of home laps or greater, sorted alphabetically by Grand Prix name.
3. Calculates the average lap time per race then saves this new information as a
7th column in file partA_output_data.txt and displays the new data.
4. Asks the user for a field to sort by and displays on screen all data contained in the file
sorted according to the user's instructions (ascending or descending).
5. Calculates the total average lap time per driver across all Grand Prix races and
presents it as a GUI column graph in a pop-up window.
6. Exit the program

Press Enter to continue...
```

Picture 4 - Result of the description function

If the user decides to exit and inputs “6” or “7” (depends on the part), the program will say goodbye to the user and stop working.

```
Enter your choice: 6
-------------------------------------------------------------------------------------------------
You selected option 6.
It's time to say goodbye then...
Bye!

Process finished with exit code 0
```

Picture 5 - Exiting the program

Datasets

For Part A, the dataset that was given was about the 2023 Formula 1 racing season. It consisted of 6 columns depicting the Grand Prix name, the date of the race, the winner name, the car team, the amount of laps and the time of the race. The data was read into the dataframe, and then multiple operations such as the calculation of average time per lap, sorting of the dataframe based on the limit of laps, sorting by multiple fields in descending and ascending orders and graph building were performed. As a result the new file was created with one extra column.

For Part B, the Bank Note Identification Dataset was chosen. Data were extracted from images that were taken from genuine and forged banknote-like specimens. It has 4 variables of indicators of the banknote image such as variance, skewness, curtosis and entropy, and the classification of each of them is genuine or forged. This data was stored in the dataframe and used to train the MLP model and then test its predictions based on some data from this dataset.

Here, the samples of input and output data is presented:

Part_B\main.py   Part_A\main.py   partA_input_data.csv ×

This file was explicitly re-assigned to plain text

```
GRAND PRIX,DATE,WINNER,CAR,LAPS,TIME
Bahrain,05-Mar-23,Max Verstappen,RED BULL RACING,57,33:56.7
Saudi Arabia,19-Mar-23,Sergio Perez,RED BULL RACING,50,21:14.9
Australia,02-Apr-23,Max Verstappen,RED BULL RACING,58,32:38.4
Azerbaijan,30-Apr-23,Sergio Perez,RED BULL RACING,51,32:42.4
Miami,07-May-23,Max Verstappen,RED BULL RACING,57,27:38.2
Monaco,28-May-23,Max Verstappen,RED BULL RACING,78,48:52.0
```

Part_B\main.py   Part_A\main.py   partA_input_data.csv   partA_output_data.txt ×

```
Bahrain,05-Mar-23,Max Verstappen,RED BULL RACING,57,33:56.7,00:35.7
Saudi Arabia,19-Mar-23,Sergio Perez,RED BULL RACING,50,21:14.9,00:25.5
Australia,02-Apr-23,Max Verstappen,RED BULL RACING,58,32:38.4,00:33.8
Azerbaijan,30-Apr-23,Sergio Perez,RED BULL RACING,51,32:42.4,00:38.5
Miami,07-May-23,Max Verstappen,RED BULL RACING,57,27:38.2,00:29.1
Monaco,28-May-23,Max Verstappen,RED BULL RACING,78,48:52.0,00:37.6
Spain,04-Jun-23,Max Verstappen,RED BULL RACING,66,27:57.9,00:25.4
Canada,18-Jun-23,Max Verstappen,RED BULL RACING,70,33:58.3,00:29.1
```

Picture 6, 7 - Input and output for Part A

Part_B\main.py   Part_A\main.py   data_banknote_authentication.txt ×

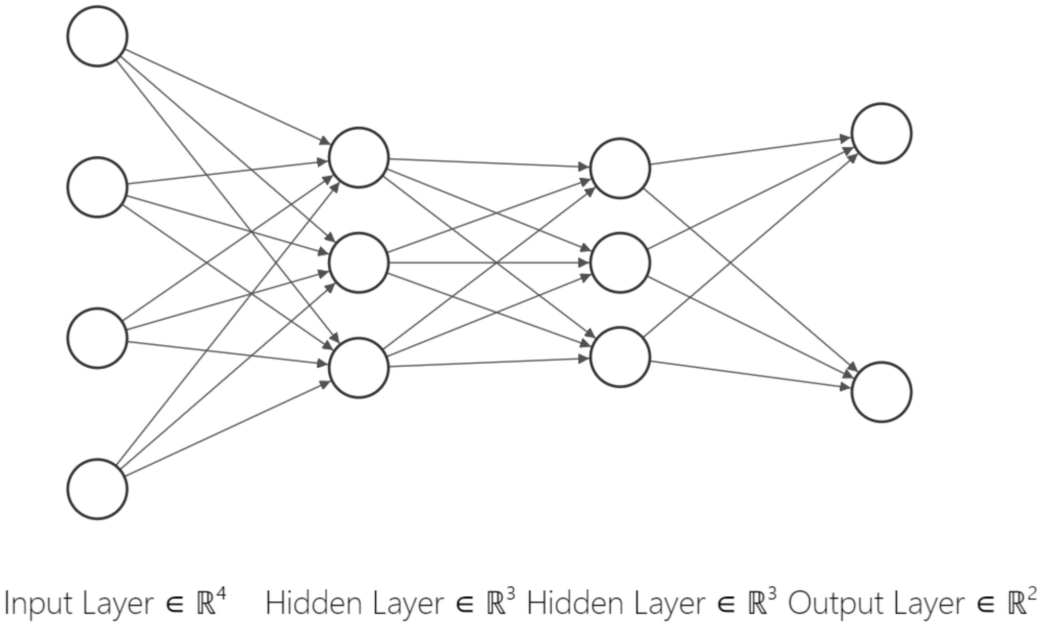
```
3.6216,8.6661,-2.8073,-0.44699,0
4.5459,8.1674,-2.4586,-1.4621,0
3.866,-2.6383,1.9242,0.10645,0
3.4566,9.5228,-4.0112,-3.5944,0
0.32924,-4.4552,4.5718,-0.9888,0
4.3684,9.6718,-3.9606,-3.1625,0
3.5912,3.0129,0.72888,0.56421,0
2.0922,-6.81,8.4636,-0.60216,0
```

```
variance,skewness,curtosis,entropy,Predictions
-3.5801,-12.9309,13.1779,-2.5677,1.0
-3.2778,1.8023,0.1805,-2.3931,1.0
3.5251,0.7201,1.6928,0.64438,0.0
2.2928,9.0386,-3.2417,-1.2991,0.0
-3.8552,3.5219,-0.38415,-3.8608,1.0
0.30081,0.17381,-1.7542,0.48921,1.0
2.3729,10.4726,-3.0087,-3.2013,0.0
5.7227,5.8312,-2.4097,-0.24527,0.0
-0.4294,-0.14693,0.044265,-0.15605,1.0
```

Picture 8, 9 - Input and output for Part B

MLP

The biggest theoretical part was dedicated towards building the MLP. As the results, the program in Part B is capable of conducting the dataset, using some part of it for training and another part for testing the MLP. Below is a ANN diagram showing the default topology:



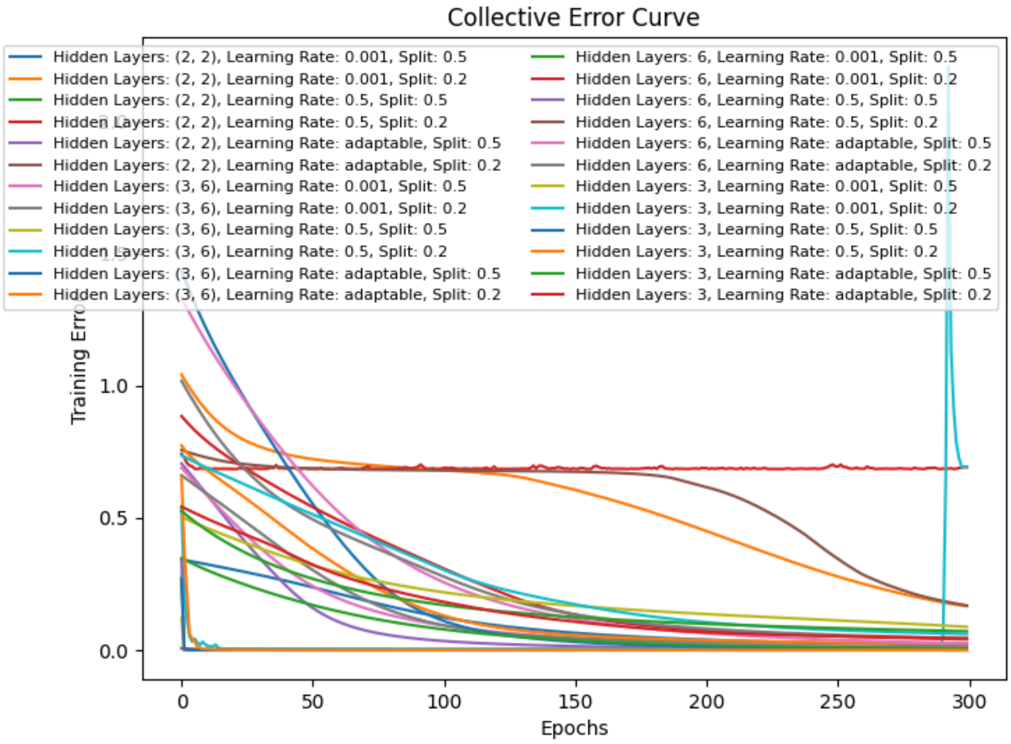
Picture 10 - ANN diagram

Since the user can choose the topology of hidden layers and the learning step, the results of the MLP can vary. The input and output layers are hard coded to fit the data requirements, so for entry nodes for each variable and two output nodes for two classes.

The 6 option of the menu does 24 different steps combining different configurations to evaluate the combination for the best performance: 4 different topologies for hidden layers

(2-2, 3-6, 6, 3), 3 different learning steps (0.001, 0.5 and adaptable), and 2 data split options (50% without shuffling and 80/20% with shuffling).

As a result, the error grapes were built for each of the combinations including the collective error graph combining all of them.



Picture 11 - Collective error curve

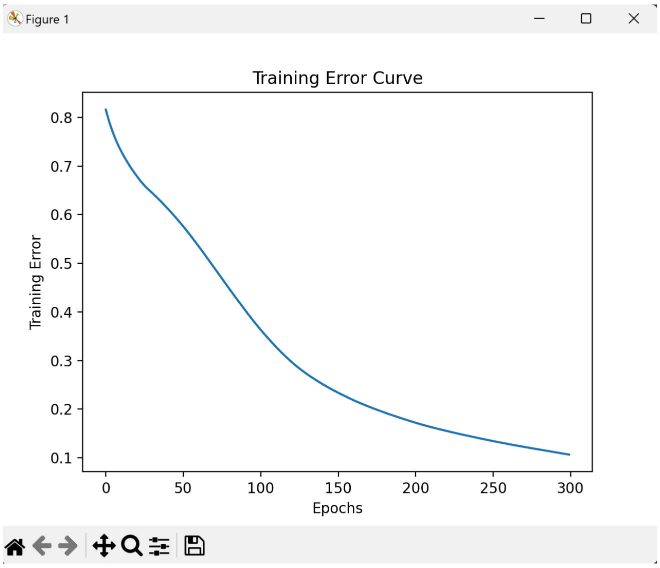
Also, some weight changing graphs were built to show the process of learning of MLP.

Picture 12, 13 - Weight changing graphs

All the graphs are built automatically and saved by the program in .png format.

As the results of testing, it seems that learning step, the topology and number of epochs play a significant role in accuracy of the model. Since the dataset was not too complicated, 300 epochs were enough in most of the cases to achieve the best result. The topology of 3-3 was also effective, putting more hidden layers or increasing the amount of nodes resulted in overfitting of the model. Considering the split option, 50/50 without shuffling tends to increase the uncertainty.

Here, the results of MLP testing with default settings:




```
Now printing the confusion matrix (without normalization)...

[[148   2]
 [  0 125]]

Now printing the classification report...

              precision    recall  f1-score   support

           0       1.00      0.99      0.99       150
           1       0.98      1.00      0.99       125

    accuracy                           0.99       275
   macro avg       0.99      0.99      0.99       275
weighted avg       0.99      0.99      0.99       275


Now printing the accuracy score...

0.9927272727272727
```

Picture 14, 15 - Error graph and classification results

Conclusion

After the work I have done, I can say that the whole project was not the easiest task. Part A was well organized to get used to the programming language and its capability to be used in further tasks. Part B turned out to be a challenging one since the results of MLP training and classification was hard to predict.

Such libraries as MatplotLib, Pandas, Numpy, Scikit-learn were very helpful and as a result of this project I learned to use them freely.

Overall, the code was built the way it can potentially run on any dataset with minimal changes.

References

1) MLPClassifier, scikit. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html (Accessed: 09 June 2024).
2) *W3schools.com*, *W3Schools Online Web Tutorials*. Available at: https://www.w3schools.com/python/ (Accessed: 09 June 2024).
3) Meyers, R.A. (1992) *Encyclopedia of Physical Science and Technology Vol. 3 CE - comm Robert A. Meyers, Ed Vol 3*. San Diego u.a.: Acad. Press.
4) *W3schools.com*, *W3Schools Online Web Tutorials*. Available at: https://www.w3schools.com/python/pandas/default.asp (Accessed: 09 June 2024).
5) *W3schools.com*, *Introduction to NumPy*. Available at: https://www.w3schools.com/python/numpy/numpy_intro.asp (Accessed: 09 June 2024).